Sophia Antipolis
October 6th & 7th 2004

Session 3B: HIGH LEVEL SYSTEM DESIGN

Multiclock Design and Synthesis with Esterel

S. Bernardi, S. Lebailly - Texas Instruments
B. Blanc, G.Berry, J.Dormoy - Esterel Technologies
Villeneuve-Loubet - France

Summary

- Objectives
- Short introduction to the Esterel language
- Multi-clock simulation model with Esterel: clocks and domain communication
- How to build the VHDL top-level
- Resynchronization circuits: theory and examples
- Applications and implementation results
- Conclusions and future steps

Objectives

- **Definition of a methodology to design multiple clock domains with Esterel, facing resynchronization issues**

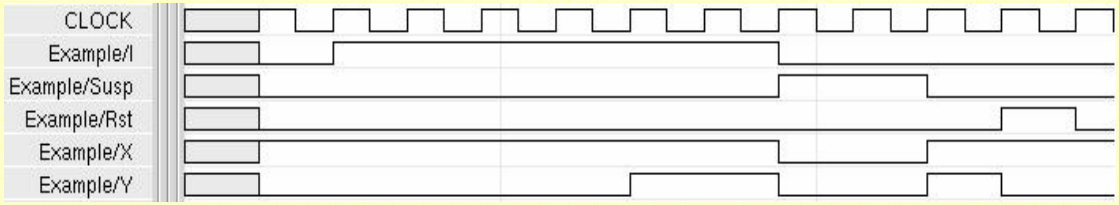
- **Generation of a synthesizable, testable and performing VHDL from the Esterel multi-clock design**

Introduction to the Esterel language and tools

What is the Esterel language and what can we do with it?

Esterel is a *high-level language* dedicated to synchronous hw and embedded sw programming.
Esterel Studio tool provides an *environment for program capture, simulation, synthesis and formal verification.*

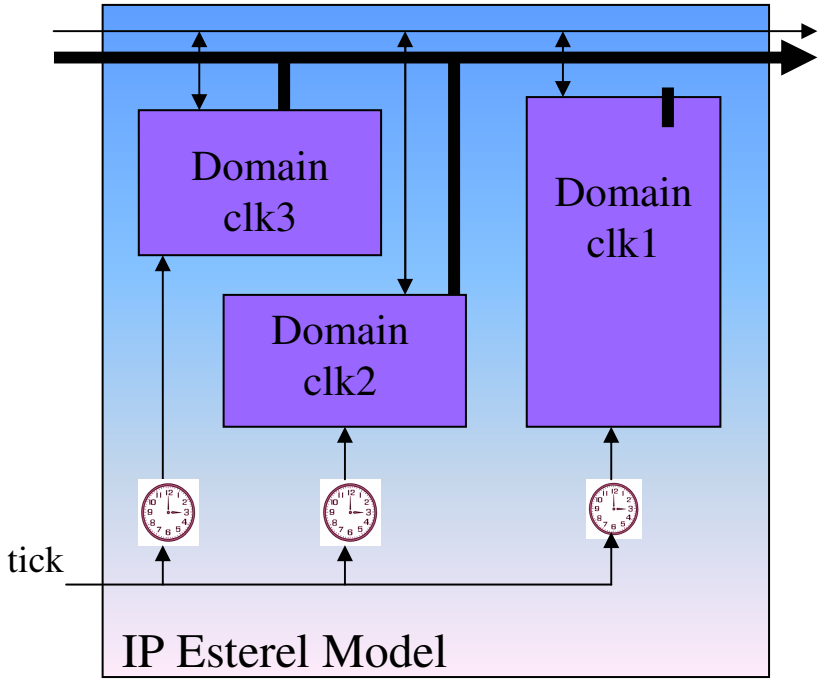
Esterel programs are built from combinationally broadcasted signals using concurrency, sequencing, pausing, preemption, and communication statements. All statements run on a unique base clock (tick), mapped to the hw clock for synchronous circuit designs.




```
module Example:
input I, Rst, Susp;
output X, Y;
loop
  suspend
    sustain X
  ||
    await 5 I;
    sustain Y
  when Susp
each Rst
end module
```

Esterel is used for control-dominated hw design by major electronic companies, but in some applications the single-clock constraint is too strong. The goal of this work is to present a practical method to extend Esterel to multiple clocks.

Multiclock simulation model in Esterel



Respect the relationship between the clocks in Esterel simulation environment, according to the real frequencies.

The input domain clocks clk1, clk2 and clk3 are generated from the global Esterel base clock.

Each domain is represented by an Esterel module constrained to work only when its clock signal is active.

Domains are directly connected to their external IOs.

Domains are connected with each other through synchronizers.

Multiclock in Esterel: implementation details

Domain clock signals can be generated by:

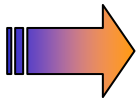
- **putting them as global inputs, and they are fully driven by the simulation environment;**
- **building them internally from the tick, through a generator with phase and period.**

```
module ClockGenerator                    :
constant Shift: integer;
constant Period: integer;
output Clock;

await Shift times tick;
 loop
        emit Clock
each (Period/2) tick

end module
```

How to use clock generators to run different clock domain modules

1: **instantiate** the clock generators

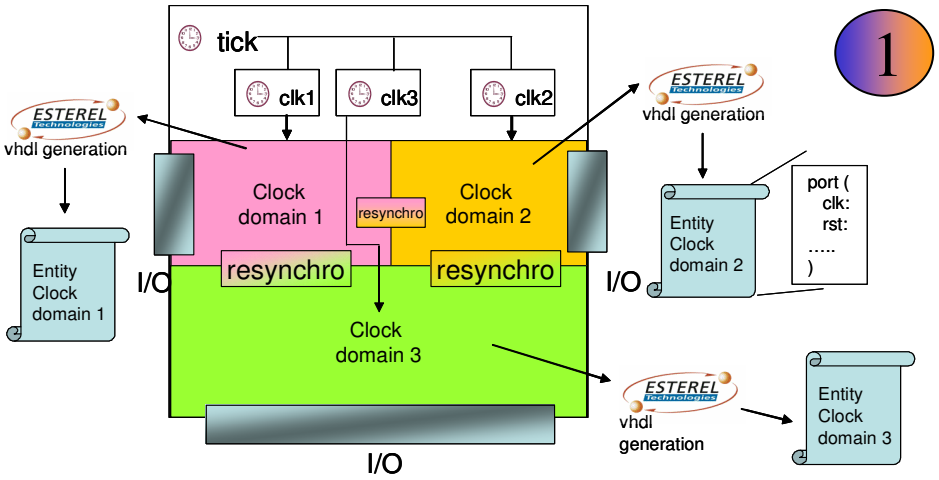
```
run ClockGenerator[
   constant 4 / Period;
   constant 2 / Shift;
   signal clk1 / Clock]
||
run ClockGenerator [
   constant 6 / Period;
   constant 3 / Shift;
   signal clk2 / Clock]
```



2: **suspend** modules when clocks absent and maintain exchanged signals in the environment

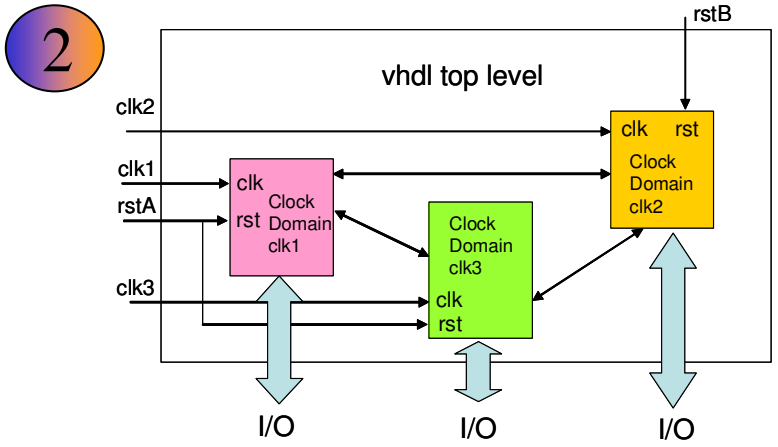
```
 suspend
    run p1 [o1 / output1]
  when immediate not clk1
||
  suspend
    run p2 [i2 / input2]
  when immediate not clk2
||
  sustain {
    i2 <= o1 and clk1,
    i2 <= pre(o1) and not clk1}
```

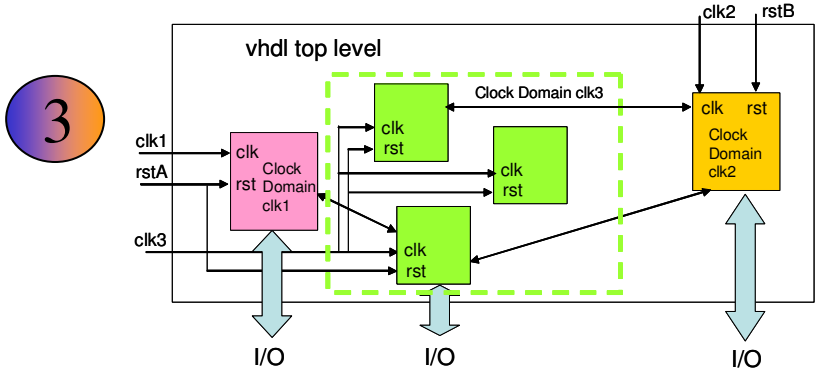
How to build the VHDL top level



The modules of each clock domains are dumped into separate HDL files.

A top level VHDL entity is manually written, instantiating the clock domain components and connecting ports, clocks and resets



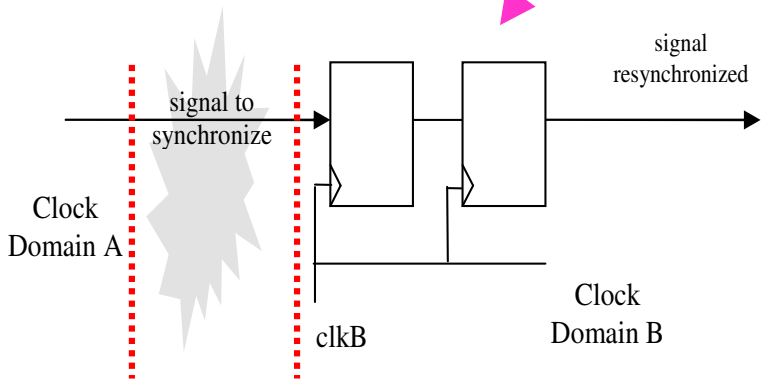


Within each clock domain, modular compiling is performed from Esterel to VHDL

Multiclock in Esterel: the resynchronization problem

A signal generated by one clock domain arrives as an asynchronous signal to the destination-clock domain, possibly violating the destination flip flop setup or hold time, and causing it to enter metastability

an hardware solution



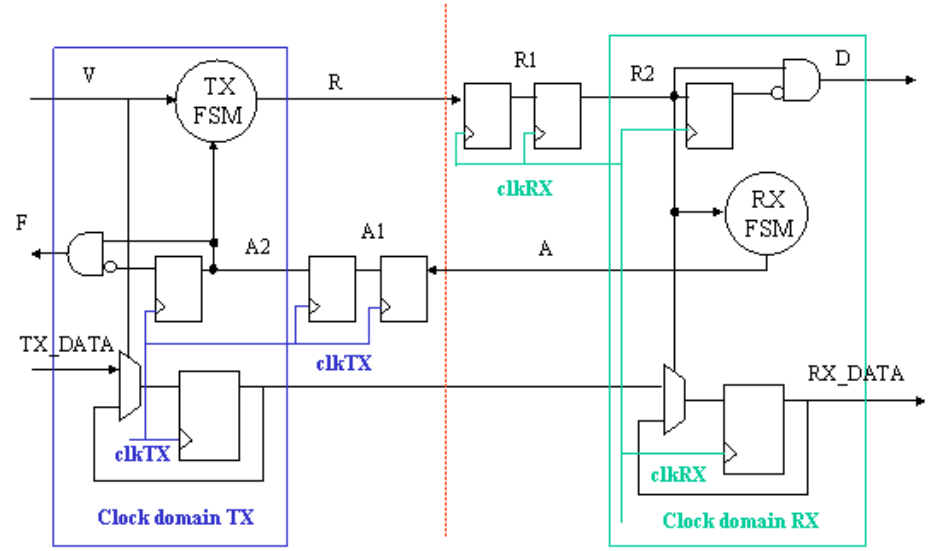
Esterel modeling

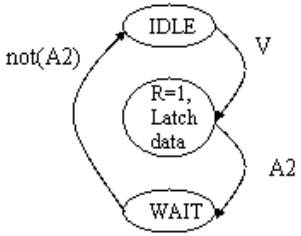
```
module resynchro:
input R;
output D: reg;
signal { R1 } : reg in
   sustain next { R1 <= R, D <= R1 }
end signal
end module
```

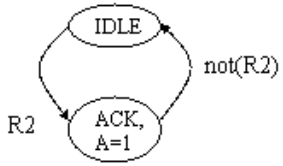
Multiclock in Esterel:
a more complex synchronization protocol

The double stage synchronizer does not ensure that the signal remains stable enough for the destination circuit to sample it once and only once.

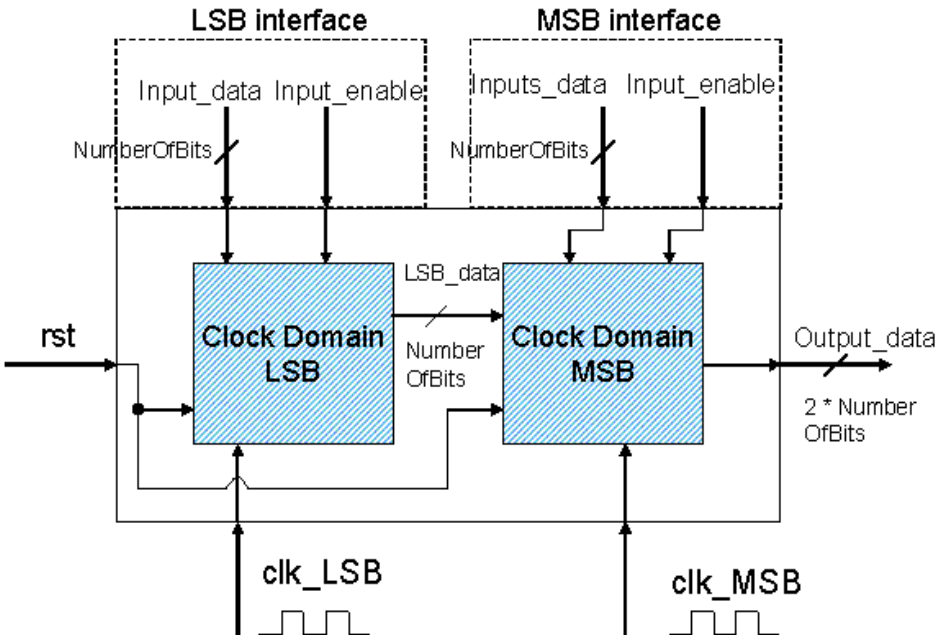
A handshake protocol such as the *push synchronizer* can solve the problem







First application: handshake protocol with a push synchronizer



Results:

Two VHDL entities generated for each domain.

Synthesis run with Synopsys Design Compiler 2003.03

No area overhead if compared with manually written VHDL.

No timing/DFT issues.

Simulations run through a VHDL test bench show a correct behavior with all foreseen frequency relationships

Second application: Video IP

Three clock domains are involved.

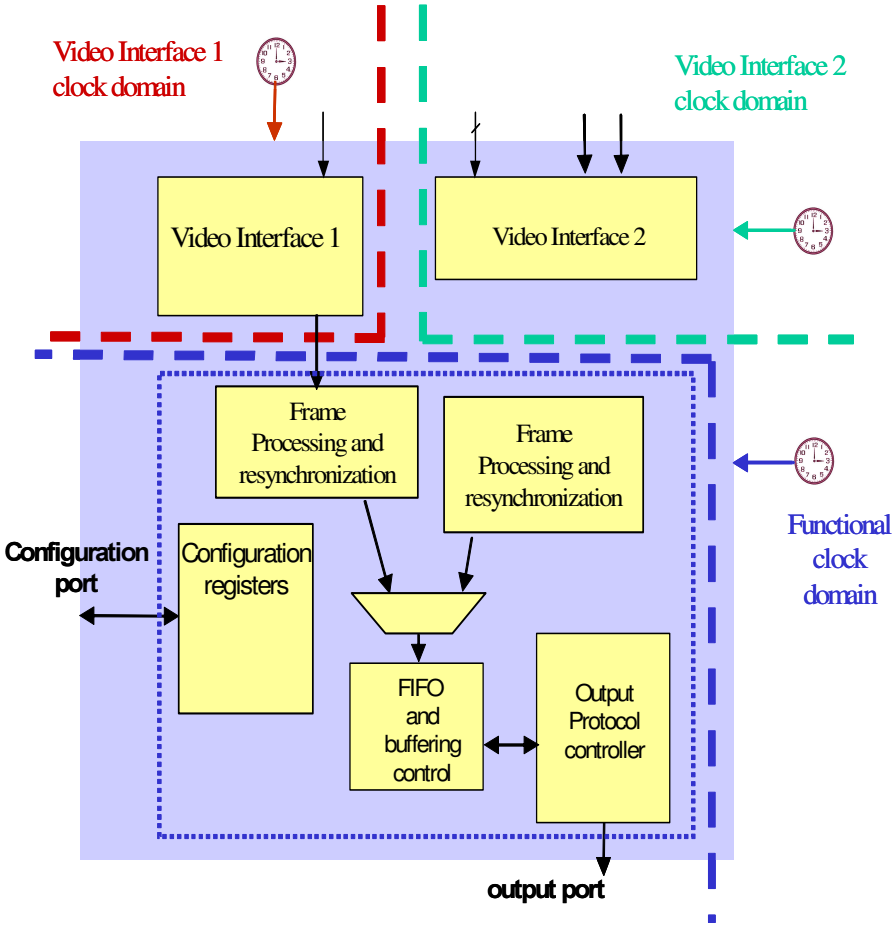
The double stage synchronizer has been use to secure data writing from the video interfaces to the buffering system.

A VHDL top-level has been created connecting the clock domain components.

Synthesis was run with Synopsys Design Compiler 2003.03 and results compared with hand-written Verilog, showing:

- no timing/DFT issues;
- a total 5% gain in area occupation.

Simulations run under the Reference Test Bench show a correct behavior



Conclusion and future steps

- A new methodology has been developed to implement multi clock design with Esterel
- Individual clock domains are modeled and synthesized using Esterel Studio tool
- For system modeling, clocks are viewed as standard signals and communication synchronizers are created from Esterel modules.
- For synthesis, individual clock domains are synthesized using the normal Esterel flow, and linked by a manually written VHDL top level.
- The overall methodology could be automatized further with upcoming versions of Esterel Studio